An Architecture for Querying Business Process, Business Process Instances, and Business Data Models

María Teresa Gómez-López[1](0000-0002-3562-875XZ) and Antonia M. Reina Quintero[1] (0000-0003-3698-6302) and Luisa Parody[1] (0000-0001-6096-8564) and José Miguel Pérez Álvarez[1] (0000-0002-0067-2662) and Manfred Reichert[2]

[1] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, Spain
{maytegomez,reinaqu,lparody,josemi}@us.es
[2] Institute of Databases and Information Systems, Ulm University, Germany
manfred.reichert@uni-ulm.de

Abstract. Business data are usually managed by means of business processes during process instances. These viewpoints (business, instances and data) are strongly related because the life-cycle of business data objects need to be aligned with the business process and process instance models. However, current approaches do not provide a mechanism to integrate these three viewpoints nor to query them all together while maintaining the information in the distributed, heterogeneous systems where they have been created. In this paper, we propose the integration of the business process, business process instance, and business data models by using their metamodels and also an architecture to support this integration. The goal of this integration is to make the most of the three models and the technologies that support them in an isolated way. In our approach, it is not necessary to change the source data formats nor transforming them into a common one. Furthermore, the proposed architecture allows us to query the three models even though they come from three different technologies.

Key words: Business Data Model, Business Process Model, Business Process Instance Model, Model Integration, Heterogeneous Data Sources

1 Introduction

The volume, variety and velocity of process-related data are growing drastically. Some of these data are created and managed by Business Process Management Systems (BPMS). A business process (BP for short) consists of a set of activities whose execution needs to be coordinated in an organisational and technical environment in order to achieve a particular business goal [24]. The coordinated execution of activities is a fundamental principle from the viewpoint of Business Process Management, and it provokes the change of data objects during

process instantiation. In general, a BP can be implemented and operationalized by a BPMS. To make the most of the heterogeneous information provided by the different viewpoints (i.e., business process, business process instance and business data), it is necessary to integrate the different models and provide a mechanism to query them all together, since they are frequently supported by different technologies. Previous solutions based on semantic models [10] have provided mechanisms to create homogeneous data stores, using the information from heterogeneous sources. The problem of this type of approaches is that they go+ against the requirements established when there are a high volume, variety and velocity of data, as in Big Data scenarios, where the changeability and quantity of data make not possible transform the sources. For this reason, this paper proposes an architecture where process querying mechanisms are inspired in the Map-Reduce paradigm. Map-Reduce is a programming paradigm that allows for massive scalability across hundreds or thousands data nodes, and it lets: (1) divide the query into the subsystems that contain the data, and (2) merge the outputs of the distributed databases.

In order to define how to divide the query and combine the obtained information, it is necessary to determine the relation between subsystems. We propose the use of metamodels and the combination of them. A metamodel defines a frame and a set of rules for creating models for a specific application domain. A viewpoint is defined in relation to one or more metamodels [9]. In our scenario, each viewpoint is supported by a metamodel that defines the main elements managed in each case and their relationships. In general, there exists a fundamental relation between the three models: Business Process Model (BPM), Business Process Instances Model (BPIM), and Business Data Model (BDM). Accordingly, it is not sufficient to query each of these models separately, but to provide integrated access to the process, data and instance perspectives. In particular, process-centric queries across these different viewpoints need to be enabled [4].

The insufficient understanding of the inherent relationships existing between business processes and business data is deficient [21]. As far as we know, existing querying approaches focus on one specific model type, but they do not exploit the information resulting from that integration. This paper shows how to integrate the BPM, BPIM, and BDM by means of a weaving model that makes explicit the relationships between the elements of the three models.

As an example of a BP considers the organisation of a conference. The corresponding BP model is shown in Figure 1. Three different pools cover the main functions needed to organize the conference, to submit a paper, and to register for the conference: (1) the first pool describes how to manage the conference organisation by the conference chair who is in charge of the conference requirements; (2) the second pool shows the submission process from the viewpoint of an author; (3) the third pool deals with the conference registration and related payment. Usually, the execution of a process instance is persisted in the database of the BPMS to which the BP is deployed. According to the framework proposed in [20], this situation corresponds to a process repository composed of

simulation models (see Def. 3.4 in [20]) This kind of information must follow the BP instance according to the used BPMS. Moreover, business data that flows through the process is persisted in a database. For the conference example, the corresponding conceptual data model is depicted in Figure 2. It presents the entities involved in the conference process. A *Conference* is defined by its *id*, *name*, *location*, *scope* (e.g., "Software Engineering", "Business Process Management", or "Big Data"), *start* and *end dates*, and the date until which the *early registration fee* is valid. In turn, an author may submit several papers, which are going to be reviewed by the *Program Committee* that decides on the *rating* and *status* of the paper.

Regarding the conference scenario, the three models (i.e., BPM, BPIM and BDM) play an important role in extracting relevant information about the business process. Examples of queries making use of the integration of these viewpoints are "Average of execution time of the conferences with scope *Business Process*", and "Evolution of the *number of registrations* variable for the process named 'Conference Management Process' of those conferences taking place in Barcelona". In particular, this paper shows how the three models may be integrated to process such queries.

The paper is organized as follows: Section 2 introduces the metamodels related to the business process, business process instance and business data viewpoints as well as the metamodel that supports the integration of them all. After that, Section 3 presents our proof-of-concept prototype. Section 4 discusses related work. Finally, Section 5 concludes the paper.

2 Business Viewpoints

This section describes the metamodels covering the three viewpoints on a BP, i.e., BPM, BPIM, BDM as well as the metamodel used to integrate them all. Then, these metamodels are used to enable an integration of the viewpoints.

A BP corresponds to a set of coordinated activities, carried out manually or automatically, to achieve a specific business goal. The metamodel for creating BP models is summarised in Section 2.1. In turn, the metamodel related to the process instance viewpoint is presented in Section 2.2. Finally, concerning

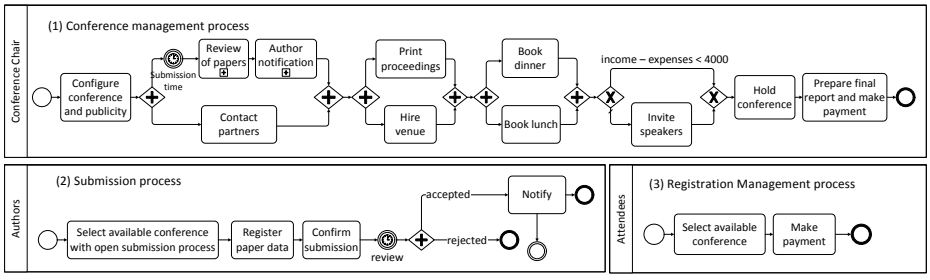

Fig. 1. Business Processes for Conference Organisations

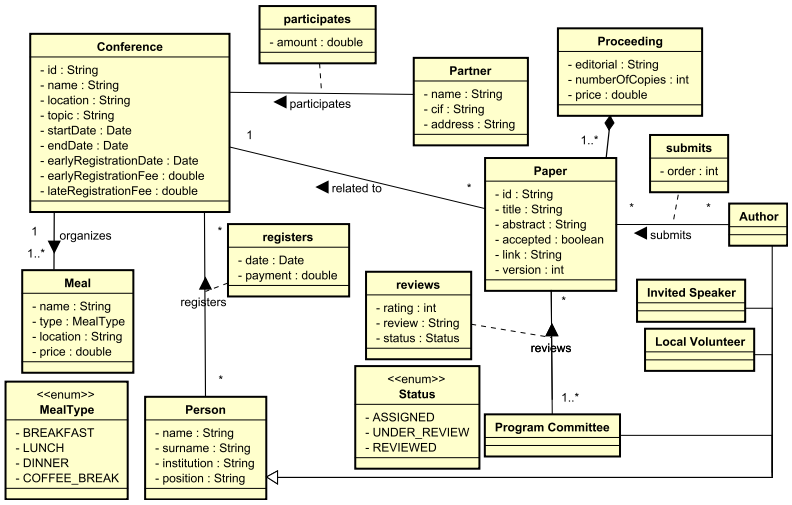

Fig. 2. Conceptual Model of the Conference Process

to the data viewpoint, a simplified metamodel of Unified Modeling Language (UML) [15] is introduced in Section 2.3. In particular, a BP needs to control its data flow, which reflects the business data managed in the context of the BP and the way this data is transferred through the activities. In general, this process-related data can be described by a conceptual model which usually is defined using UML. In practice, the three models must be considered in an integrated way due to their many interdependencies; e.g., BP instances are related to a BP model, data entities are used to define the data flow of a BP, and the various BP instances create data objects or update the values of the data.

2.1 Viewpoint 1: Business Process Metamodel

The main standard used to model business processes is the Business Process Model and Notation (BPMN) as proposed by OMG [14]. Besides other elements, BPMN 2.0 includes data objects, events of various types, and artefacts. Figure 3 introduces a simplified version of the BP metamodel (BPMM for short), which includes the main metaclasses used in this paper to model the BPs as well as their relationships and attributes.

The root of the metamodel is `BPMNProcess`, which is composed of a set of `BPMNElement`. A `BPMNElement`, in turn, may be a `Swimlane`, a `ConnectingObject`, a `FlowObject`, and an `Artifact`. Note that these metaclasses are abstract. The concrete metaclasses are subtypes of these ones. Thus, a `ConnectingObject` may be a `MessageFlow`, a `SequenceFlow`, and an `Association`. Furthermore, there are three types of `FlowObject`: `Event`, `Gateway` and `Activity`. The three of them are also abstract metaclasses. An `Event` may be an `InitialEvent`, an `IntermediateEvent`, and a `FinalEvent`. A `Gateway` may be a `XOR`, `OR`, and `AND` gateway. An `Activity`

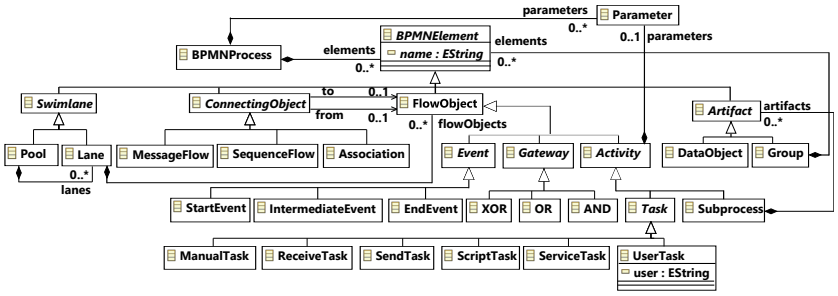

Fig. 3. Simplified business process metamodel

may be a `Task` (also an abstract metaclass) or a `Subprocess`. There are different types of `Task`: `ManualTask`, `ReceivedTask`, `SendTask`, `ScriptTask`, `ServiceTask`, and `UserTask`. Note that the semantics of these metaclasses is exactly the specified by the BPMN2.0 standard and also that this metamodel is a simplified version of it. We refer interested readers to [14] for knowing the semantics of each metaclass.

2.2 Viewpoint 2: Business Process Instance Metamodel

A business process instance represents a concrete case in the operational business of a company [24]. Figure 4 shows our proposal of Business Process Instance Metamodel (BPIMM for short). `ProcessEngine` is the root of the metamodel and it represents the Business Process Management System (BPMS) that executes the BP. A BPMS may deploy various processes. Each process is described by means of the `ProcessDefinition` metaclass. Note that `ProcessDefinition` is related to a Business Process (cf. Section 2.1). `ProcessInstance` represents an execution of a specific process. Thus, a `ProcessDefinition` is related to a set of `ProcessInstance`. A `ProcessDefinition` is composed of a sequence of `Activity`. Accordingly, a `ProcessInstance` is composed of a set of `ActivityInstance` (`childActivities` relation). There is also a parent-child relation between activities instances. The first activity that is instantiated during a process execution is an activity instance that has no parent. Finally, a set of `Variable` can be associated to a `Process Instance`.

Note that BPIMM deals with concepts related to process execution and that the definition of processes included in many BPMSs usually has more properties of processes and activities than the standard BPMN 2.0. Having this into account, the ProcessDefinition and Activity metaclasses model these properties that are present in the execution environments.

2.3 Viewpoint 3: Business Data Metamodel

In order to be able to create BDM, we use the OMG Metamodel to represent Conceptual Models introduced in Figure 5, which includes the main entities of the Business Data metamodel (BDMM) presumed in this paper. The root entity is

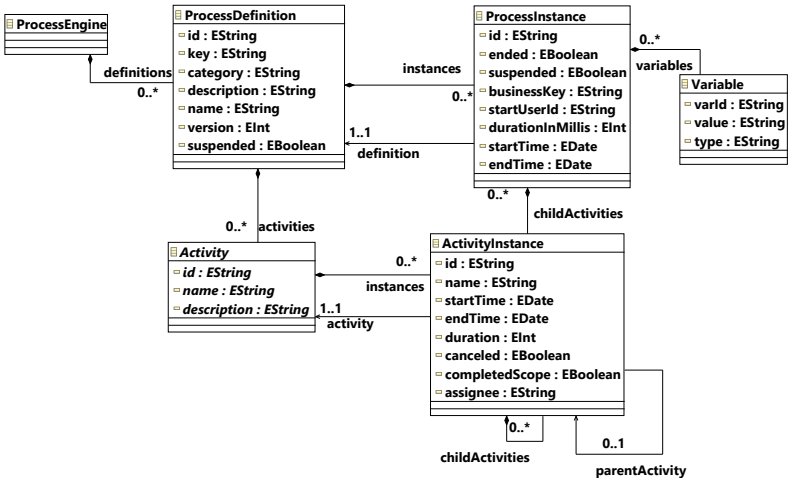

Fig. 4. Business Process Instance metamodel (BPIMM)

`BMDiagram`, which is composed of a set of `ModelElement`. A `ModelElement` may be an `AssociationEnd`, a `Relationship`, a `Classifier`, a `TypedElement`, or a `Feature`. The entities of the system are mainly represented by a `Classifier`, either an `Association`, `Class`, or `DataType`. Furthermore, through the `AssociationEnd` and `Relationship`, `Classifier` entities are related. Note that an instance of this metamodel which depicts entities related to the organisation of a conference is presented in Figure 2.

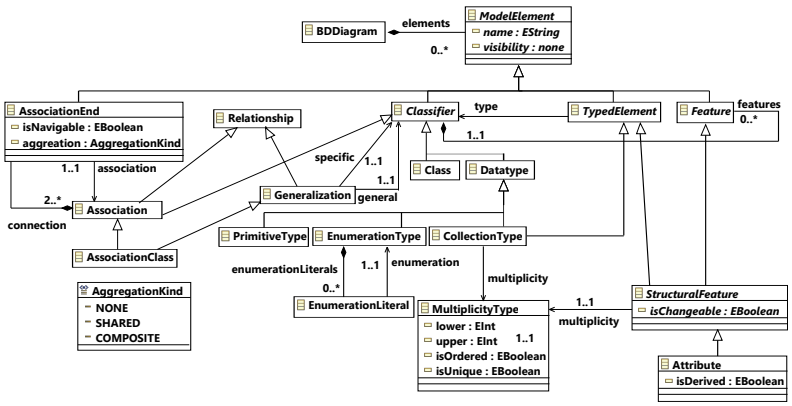

Fig. 5. Simplified Business Data Metamodel (BDMM)

2.4 Integrating BPM, BPIM and BDM

Business process, business process instance and business data metamodels represent different, but complementary, viewpoints of a business process. Linking entities across these metamodels is a must in order to exploit the information provided by their specific instances, i.e., their models. In this context, model weaving is a generic operation that establishes correspondences between model elements. The resulting weaving models are a special kind of models that link together other models. In general, weaving models contain a set of links between the elements of two different models [8]. A weaving model conforms to a weaving metamodel, which defines the kind of links that may be established between the elements of the woven models.

Atlas Model Weaver (AMW) [8] is a tool that allows creating and handle weaving models and metamodels. New weaving metamodels should be implemented by extending a Core Weaving Metamodel that it is included and implemented in AMW. Thus, to define our integration metamodel we extend this Core Weaving Metamodel. Figure 6 shows our extension (see the metaclasses with grey background colour), which specifies the semantics of the links between the elements of Business Process, Business Process Instance and Business Data models. These elements are specific instances of the corresponding metamodels (i.e., BPMM, BDMM, and BPIMM). Note that classes belonging to the core weaving metamodel are abstract, and are extended by metaclasses (the ones coloured in grey) that refine them by adding the semantics needed in the context of our scenario.

An Integration Model can be seen as a kind of weaving model that let us integrate the three kinds of models (BPIM, BDM, and BPM). This integration model defines the relationships between the elements in the three models. Match is a kind of link representing the relation between two elements from different models.

Figure 7 shows how three instances (i.e., models) of BPIMM, BPMM and BDMM may be related by means of an instance of the Integration Metamodel. Classes with <<BPIM>> stereotype represent instances of Business Process Instance metaclasses, classes with <<BPM>> represent instances of Business Process metaclasses, classes with <<BDM>> stereotype represent instances of Busi-

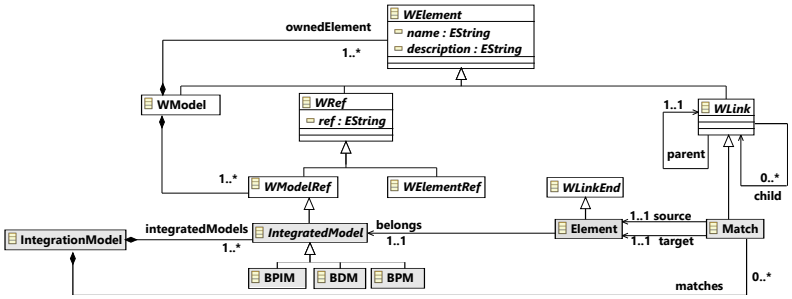

Fig. 6. Integration metamodel

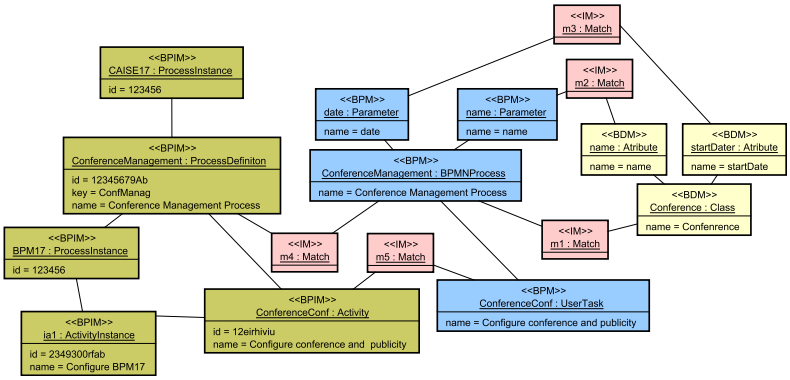

Fig. 7. Integrated BPIM, BPM and BDM

ness Data metaclasses, and classes with `<<IM>>` represent instances of Integration metaclasses. Thus, the `ConferenceManagement::BPMNProcess` instance represents the process named Conference Management Process, as depicted in Figure 1, whereas the instance `ia1::ActivityInstance` represents one execution of the activity that configures the Conference BPM17.

3 Architecture for Combined Queries

Data combination is an important open problem in business intelligence, as explained in [20], where the authors propose a general, abstract framework for devising process querying methods. Our architecture, based on Map-Reduce paradigm, defines how the query is divided (Map), and the partial queries are combined (Reduce). Our proposal can be seen as an instance of the abstract framework presented in [20], in such a way that it includes a real combination of technologies that can be thought as the implementation of some of the generic functionalities described in the framework. Assume one wishes to know how many institutions are involved in the execution of each task, in a context where the BPMS Bonita^{TM}, data are persisted in an Oracle database and the BPM is persisted by using Neo4J a query similar to the one shown in Listing 1 should be executed:

```
SELECT Task.id, COUNT(institution)
FROM (Task FULL JOIN ActivityInstance FULL JOIN Person)
GROUP BY Task.id WHERE Task.idProcess = idP
```

Listing 1. Query example

Note that obtain the query results, you need to query data related to the three viewpoints. In other words, you need information about Tasks from the BP viewpoint, information about Activity Instances from the BPI viewpoint, and information about Persons from the BD viewpoint. Note that to obtain this information we have to deal with three different technologies (Neo4J, Bonita, and

Oracle, respectively). We propose a two-step process to execute the query aligned with Map-Reduce methods. The first step, data extraction, is the responsible for transforming the original query into a set of queries that are specific for each viewpoint. The second step, data integration, has as input the results of the previous queries and is the responsible for combining them. Figure 8 shows the architecture that supports the two-steps query execution. The following sections give some details about these two steps.

3.1 Data Extraction

In the data extraction step, the original query is transformed into three different queries, in order to query the three different viewpoints in an isolated way (see (1) in Figure 8). The result of these specific queries (step 2) is a set of JSON files that will be combined in step (3). The interaction with the specific technology that supports each viewpoint (i.e., Neo4J, Bonita and Oracle) is in charge of drivers. Drivers are the components that deal with the platform specific features of each technology. In other words, drivers transform a generic query to a specific query on the respective data repository. Note that these three queries can be executed in parallel since each result is obtained from isolated databases.

All drivers should implement the generic operation select. This select operation extracts a set with all of the objects of a given class that fulfil certain predicate. That is, every driver should implement an operation with a prototype similar to the following one: `Set<T> select (Class<?> name, Predicate<T> pred)`. Thus, our original query (see Listing 1) is transformed into three select calls (one to obtain a set of tasks, another one to obtain a set of Activity Instances, and, finally, one to obtain a set of Persons) that will be executed by one driver each.

The Neo4J Driver transforms the select operation into a Neo4J query to obtain the set of tasks of the process whose id is `idP`. The driver uses Spring Data and neo4j-ogm-bolt-driver to map the graph database into Java Objects. The Bonita Driver transforms the select operation into a set of calls to the BONITA REST API to get the set of Activity Instances. The Oracle Driver transforms the select operation into an Oracle query to obtain the set of Persons stored in the database.

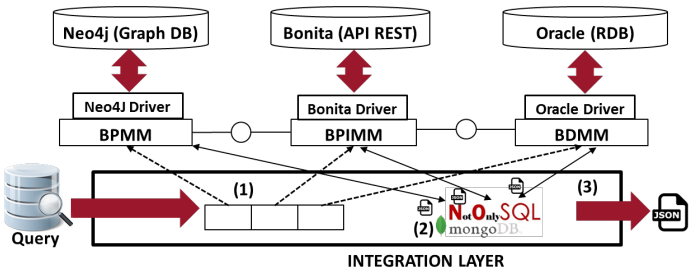

Fig. 8. Combination of Technologies for the Architecture

3.2 Data Integration

In the data integration step, the results are integrated to obtain the result of the generic query. Our approach uses *JSON* as data interchange mechanism, in such a way that the *JSON* files can be uploaded in the end-point and then be combined using the MongoDB query language. Thus, full join and aggregation operations allow us to combine the partial solutions in a single one.

The service that manages the specific queries to the various drivers, stores the results of each specific query in a centralised MongoDB. The query process in the MongoDB is a Spring application. This service further uses Spring Data MongoDB in order the manage data in the database.

Although the proposed architecture is a step beyond BI data extraction [20], some limitations are still a challenge: do the current technologies support the attributes defined in each metamodel, how can affect the query operations supported by subsystems to the development of the drivers, and can the integration layer apply any query operation to combine subsets of data.

4 Related Work

The approaches related to our work can be classified into two different groups:

Approaches that specify a viewpoint or a combination of viewpoints. Our contribution in this area is the Business Process Instance metamodel, as our business process and business data metamodels are a simplification of the corresponding standards proposed by OMG, BPMN2.0 [14] and UML2.0 [15]. As a consequence, this group of approaches focuses only on those ones that deal with the modelling of process instances. In relation to this, the necessity to store the history of process instances has been analysed in [13] and in [17]. However, there is no standard definition of the process instance metamodel. Frequently, ad-hoc models are used according to the BPMS that supports the solution. In [18] a multi-view metamodel is proposed for business process instance model. The metamodel deals with three different viewpoints: process execution path, process instance data and process instance metadata.

The relation between data stored and process activities have been studied in [11], [6] and [12], but not having as a goal to query both at the same time, only to have a way to create a model where both aspects are combined.

Approaches for querying the viewpoints. Approaches that query the different viewpoints can be classified according to their capacities to query data into three groups: those that can query only a viewpoint in an isolated way, those that can query a combination of two of the viewpoints, and, finally, those that can query the combination of all of them.

In the first group, there are approaches such as BP-QL [1], BPMN-Q [22], APQL [23] that only query the business process viewpoint or IPMPQL [5], BeehiveZ [16], BP-Mon [2] and BP-SPARQL [3] only query the business process

instance viewpoint. In the second group, is PIQL [19] that queries, on the one hand, the business process viewpoint, and, on the other hand, the combination of the business process instances and data viewpoints. Data-Aware POQL [7] also belongs to the second group by querying the combination of Business Process Instance and Business Data viewpoints. Finally, note that none of the proposals allows us to combine the three models in the same query by using the same language, while our proposal allows any combination of metamodels in a query.

5 Conclusions

Business process models are usually executed in a Business BPMS, which registers each execution of the model, the so-called business process instance. Furthermore, the data that flows through the process is persisted in an external database belonging to the company. Thus, business processes can be viewed from three different perspectives or viewpoints, namely, business process, business process instance and business data. In this paper, we specify the elements that are involved in each viewpoint by means of a metamodel. The relations between the models that represent the different viewpoints are made explicit by means of an integration model that also conforms to an integration metamodel. The integration model is a kind of weaving model that specifies the semantics of the relationships between the elements that belong to the different viewpoints.

This paper presents an architecture that allows us to extract in a combined way the different parts of the business process knowledge, providing two main benefits: (1) no data source modifications must be done, evaluating the subqueries into each data repository, reducing the time and complexity of data transformation; and, (2) an architecture for a possible combination of concrete technologies has been proposed.

Acknowledgment

This work has been partially funded by the Ministry of Science and Technology of Spain (TIN2015-63502-C3-2-R, TIN2013-40848-R and TIN2016-75394-R) and the European Regional Development Fund (ERDF/FEDER).

References

1. C. Beeri, A. Eyal, S. Kamenkovich, and T. Milo. Querying business processes with BP-QL. *Inf. Syst.*, 33(6):477–507, 2008.
2. C. Beeri, A. Eyal, T. Milo, and A. Pilberg. BP-Mon: query-based monitoring of BPEL business processes. *SIGMOD Record*, 37(1):21–24, 2008.
3. S.-M.-R. Beheshti, B. Benatallah, and H. R. M. Nezhad. Enabling the analysis of cross-cutting aspects in ad-hoc processes. In *CAiSE*, pages 51–67, 2013.
4. S.-M.-R. Beheshti, B. Benatallah, S. Sakr, D. Grigori, H. R. Motahari-Nezhad, M. C. Barukh, A. Gater, and S. H. Ryu. *Process Analytics - Concepts and Techniques for Querying and Analyzing Process Data*, chapter 1. Business Processes: An Overview, pages 1–18. Springer, 2016.

5. I. Choi, K. Kim, and M. Jang. An XML-based process repository and process query language for integrated process management. *Knowledge and Process Management*, 14(4):303–316, 10 2007.
6. E. G. L. de Murillas, H. A. Reijers, and W. M. P. van der Aalst. Connecting databases with process mining: A meta model and toolset. In *BMMDS/EMMSAD*, pages 231–249, 2016.
7. E. G. L. de Murillas, H. A. Reijers, and W. M. P. van der Aalst. Everything you always wanted to know about your process, but did not know how to ask. In *BPM Workshops 2016*, pages 339–351, 2016.
8. M. D. D. Fabro and P. Valduriez. Towards the efficient development of model transformations using model weaving and matching transformations. *Software and System Modeling*, 8(3):305–324, 2009.
9. K. Fischer, D. Panfilenko, J. Krumeich, M. Born, and P. Desfray. Viewpoint-based modeling-towards defining the viewpoint concept and implications for supporting modeling tools. In *EMISA*, pages 123–136, 2012.
10. C. D. Francescomarino, F. Corcoglioniti, M. Dragoni, P. Bertoli, R. Tiella, C. Ghidini, M. Nori, and M. Pistore. Semantic-based process analysis. In *International Semantic Web Conference (2)*, pages 228–243, 2014.
11. M. T. Gómez-López, D. Borrego, and R. M. Gasca. Data state description for the migration to activity-centric business process model maintaining legacy databases. In *BIS*, pages 86–97, 2014.
12. M. T. Gómez-López, J. M. Pérez-Álvarez, Á. J. Varela-Vaca, and R. M. Gasca. Guiding the creation of choreographed processes with multiple instances based on data models. In *BPM Workshops 2016*, pages 339–351, 2016.
13. K. Grigorova and I. Kamenarov. Object relational business process repository. In *CompSysTech*, pages 72–78, 2012.
14. O. M. Group. *Business Process Model and Notation (BPMN) Version 2.0*. OMG Standard, 2011.
15. O. M. Group. *Unified Modeling Language Reference Manual, Version 2.5*. OMG Standard, 2015.
16. T. Jin, J. Wang, M. L. Rosa, A. H. M. ter Hofstede, and L. Wen. Efficient querying of large process model repositories. *Computers in Industry*, 64(1):41–49, 2013.
17. Z. Ma, B. Wetzstein, D. Anicic, S. Heymans, and F. Leymann. Semantic business process repository. In *SBPM*, 2007.
18. N. N. Moghadam and H.-Y. Paik. BPIM: A multi-view model for business process instances. In *APCCM*, pages 23–32, 2015.
19. J. M. Pérez-Álvarez, M. T. Gómez-López, L. Parody, and R. M. Gasca. Process instance query language to include process performance indicators in DMN. In *EDOC Workshops 2016*, pages 1–8, 2016.
20. A. Polyvyanyy, C. Ouyang, A. Barros, and W. M. P. van der Aalst. Process querying: Enabling business intelligence through query-based process analytics. *Decision Support Systems*, pages –, 2017.
21. M. Reichert. Process and data: Two sides of the same coin? In *OTM Conferences (1)*, pages 2–19, 2012.
22. S. Sakr and A. Awad. A framework for querying graph-based business process models. In *WWW*, pages 1297–1300, 2010.
23. A. H. M. ter Hofstede, C. Ouyang, M. L. Rosa, L. Song, J. Wang, and A. Polyvyanyy. APQL: A Process-Model Query Language. In *AP-BPM*, pages 23–38, 2013.
24. M. Weske. *Business Process Management: Concepts, Languages, Architectures*. Springer, 2007.